World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:18, No:4, 2024

Experimental Study of Hyperparameter Tuning a Deep Learning Convolutional Recurrent Network for Text Classification

Bharatendra Rai

Abstract**—**Sequences of words in text data have long-term dependencies and are known to suffer from vanishing gradient problem when developing deep learning models. Although recurrent networks such as long short-term memory networks help overcome this problem, achieving high text classification performance is a challenging problem. Convolutional recurrent networks that combine advantages of long short-term memory networks and convolutional neural networks, can be useful for text classification performance improvements. However, arriving at suitable hyperparameter values for convolutional recurrent networks is still a challenging task where fitting of a model requires significant computing resources. This paper illustrates the advantages of using convolutional recurrent networks for text classification with the help of statistically planned computer experiments for hyperparameter tuning.

Keywords**—**Convolutional recurrent networks, hyperparameter tuning, long short-term memory networks, Tukey honest significant differences

I. Introduction

ANALYSIS of text data that belong to an unstructured category of data is challenging due to their unique nature involving a sequence of words. Developing deep network models for text data such as customer comments, product reviews, movie reviews, etc., is an active area of research. One of the deep learning networks, known as Long Short-Term Memory (LSTM) networks, is a special type of Recurrent Neural Network (RNN) that proves valuable for handling sequential data and offers specific advantages. A key advantage of using LSTM networks lies in the fact that it addresses the 'vanishing gradient problem' that makes network training difficult for long sequences of words or integers [1]. Gradients are employed to update RNN parameters, but for lengthy sequences of words or integers, these gradients gradually diminish to the point where network training becomes ineffective. Although LSTM networks help solve an important problem, it is still challenging to achieve high model performance with text data.

Deep learning models are an active area of research and there are a number of applications to a wide variety of problems [2]-[5]. A well-known deep learning network, called convolutional neural networks (CNN), has been found to be useful in capturing high level local features from data. Reference [6] used a CNN model with transfer learning for a scene script identification problem. Reference [7] used a CNN model for a large-scale image recognition problem. Reference [8] used 1D CNN for developing an automatic emotion recognition model from speech data. Reference [9] presented an analysis into the inner workings of CNNs for processing text data.

Deep learning models, where both CNN and LSTM neural networks are used in the same model architecture, are called as convolutional recurrent neural networks (CRNN). CRNN models have also been applied to a wide variety of research problems [10]-[15]. This paper focuses on CRNN deep learning models. Although using CRNN models can be useful for handling text classification problems, arriving at suitable values for hyperparameters related to such models requires a systematic approach due to the high number of parameters involved and related need for computing resources when training such a deep learning model.

In this paper, we make use of statistically designed experiments to identify statistically significant hyperparameters and to arrive at their suitable values to improve model performance. For text data, we will make use of reuter_50_50 dataset available from UCI Machine Learning Repository. This data contains text files in two folders with one folder each for train and test data. The folder containing training data has 2500 text files with 50 articles each from 50 different authors. Similarly, the folder containing test data also have 2500 text files with 50 articles each from the same 50 authors that appear in the training data. We will develop a CRNN model for author classification using the training data and assess model performance using the test data.

II. Data Preparation for Model Building

Preparation of text data for developing the CRNN model is done by converting the articles in text form to a sequence of integers. During this tokenization process, the number of most frequent words used in this study is 1500. Similarly, labels involving author names are also converted into integers. For developing the author classification model, the number of integers for each of the train and test text data needs to be of equal length. This is achieved by padding and truncation of the sequence of integers. In this study, maximum length of words used in the train and test data is 400. When the sequence of integers is smaller than 400, extra zeroes are added to the sequence of integers to artificially increase the number of

B. Rai is with the University of Massachusetts - Dartmouth, Dartmouth, MA 02747 USA (phone: 508-910-6434; e-mail: brai@umassd.edu).

integers to 400. This operation is called as padding. Similarly, when the sequence of integers is larger than 400, the extra integers are deleted. This operation is called truncation. Thus, padding and truncation help to equalize the number of integers in the train and the test data.

Next, the train data are randomly partitioned into training and validation data containing 80% and 20% respectively of the train data. The training data are used for fitting the model and the validation data are used for assessing the model during each iteration of the learning process. Finally, one-hot encoding of labels in the training, validation and test data is carried out. With this last step, the data are ready for developing a convolutional recurrent network model for author classification based on the articles they have written.

This study used R programming for data preparation and all subsequent analysis and visualization. Two key libraries used are Keras and TensorFlow.

III. Convolutional Recurrent Network Model Architecture

The convolutional recurrent network architecture is captured in the form of a simple flowchart in Fig. 1.

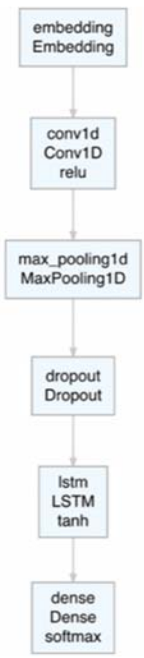

Fig. 1 Layers in the convolutional recurrent network architecture

As shown in Fig. 1, convolutional recurrent network architecture in this study consists of embedding, convolutional 1D, maximum pooling, LSTM and dense layers. In the convolutional layer 'relu' activation function is used. In LSTM and dense layers 'tanh' and 'softmax' activation functions respectively are used. The model architecture also includes pooling and dropout layers that help to avoid overfitting. Reference [16] provides a detailed coverage of various layers in a deep learning network. Table I provides a summary of convolutional recurrent network architecture with output shape and number of parameters related to each layer as an example.

TABLE I
Summary of a Convolutional Recurrent Network Architecture

Layer (type)	Output shape	Number of parameters
embedding (Embedding)	(None, 400, 32)	48000
conv1d (Conv1D)	(None, 198, 128)	20608
max_pooling1d (MaxPooling1D)	(None, 66, 128)	0
dropout (Dropout)	(None, 66, 128)	0
lstm (LSTM)	(None, 64)	49408
dense (Dense)	(None, 50)	3250
Total parameters		121,266

For compiling the model, the optimizer is specified as 'adam'. In this study, since labels are based on 50 authors, 'categorical_crossentropy' is used as loss function. And for metrics, accuracy of author classification is specified for assessing network performance.

IV. Hyperparameter Tuning Experiment

For hyperparameter tuning experiment to develop the author classification model, five factors are identified from the convolutional recurrent network architecture. The factors and their levels used in the experiment are summarized in Table II.

TABLE II
Factors and Levels Used in the Hyperparameter Tuning Experiment

Factors	Level-1 (low)	Level-2 (medium)	Level-3 (high)
Filters in CNN layer (cnn)	32	64	128
Kernel size (kernel)	4	-	5
Strides of convolution (strides)	1	-	2
Dropout layer rate (drop)	0.1	0.2	0.3
Units in LSTM layer (lstm)	16	32	64

The number of filters in the convolutional layers is set at three levels: 32, 64, and 128. The kernel size determines the length of the 1D convolution window, and for this experiment, levels 4 and 5 are utilized. Strides indicate the stride length of the convolution and are tested at two levels: one and two. A dropout layer, which aids in mitigating the overfitting problem, is configured at levels of 10%, 20%, and 30%. The number of units in the LSTM layer is set at three levels: 16, 32, and 64. Therefore, this hyperparameter tuning experiment consists of three factors at three levels and two factors at two levels, resulting in a total of 108 trials (3 x 3 x 3 x 2 x 2 = 108) for a full-factorial design. To expedite these experiments, they were conducted on Dell's Mobile Data Science Workstation Precision 7750 equipped with an NVIDIA Quadro RTX 5000 graphics processing unit (GPU). Leveraging GPU computing significantly reduced the time required to complete all the experimental runs, from days to hours.

V. Results and Model Assessment

For fitting the model, number of epochs was kept constant at 30 for all combinations in the experiment. For each of the 108 experimental combinations, accuracy of author classification based on validation data is used as response. Data obtained from the experiment were analyzed using analysis of variance method and the resulting information is summarized in Table III.

TABLE III
ANOVA TABLE USING DATA FROM THE HYPERPARAMETER TUNING EXPERIMENT

Source of variation	Degrees of freedom	Sum of squares	Mean square	F value	p-value
Filters in CNN layer (cnn)	2	0.06941	0.03471	22.007	1.17e-08
Kernel size (kernel)	1	0.01508	0.01508	9.563	0.00257
Strides of convolution (strides)	1	0.03064	0.03064	19.427	2.61e-05
Units in LSTM layer (lstm)	2	0.11436	0.05718	36.259	1.35e-12
Residuals	101	0.15928	0.00158		

Out of the five factors used in the experiment, four factors were found to be statistically significant. The main effect of the dropout layer rate was not found to have a statistically significant impact on author classification. Interaction effects were also not found to be statistically significant. The main effect of the dropout layer rate and interaction effects were combined with residuals, resulting in 101 degrees of freedom. To arrive at the best levels of the statistically significant factors, Tukey honest significant differences based on Studentized range statistics were computed [17], [18]. These confidence intervals are based on the range of the sample means rather than individual differences. They help overcome the shortcoming of simple comparisons using t-tests, which can inflate the probability of declaring a significant difference when it is not actually present. The resulting confidence intervals for the differences between means are shown in Fig. 2.

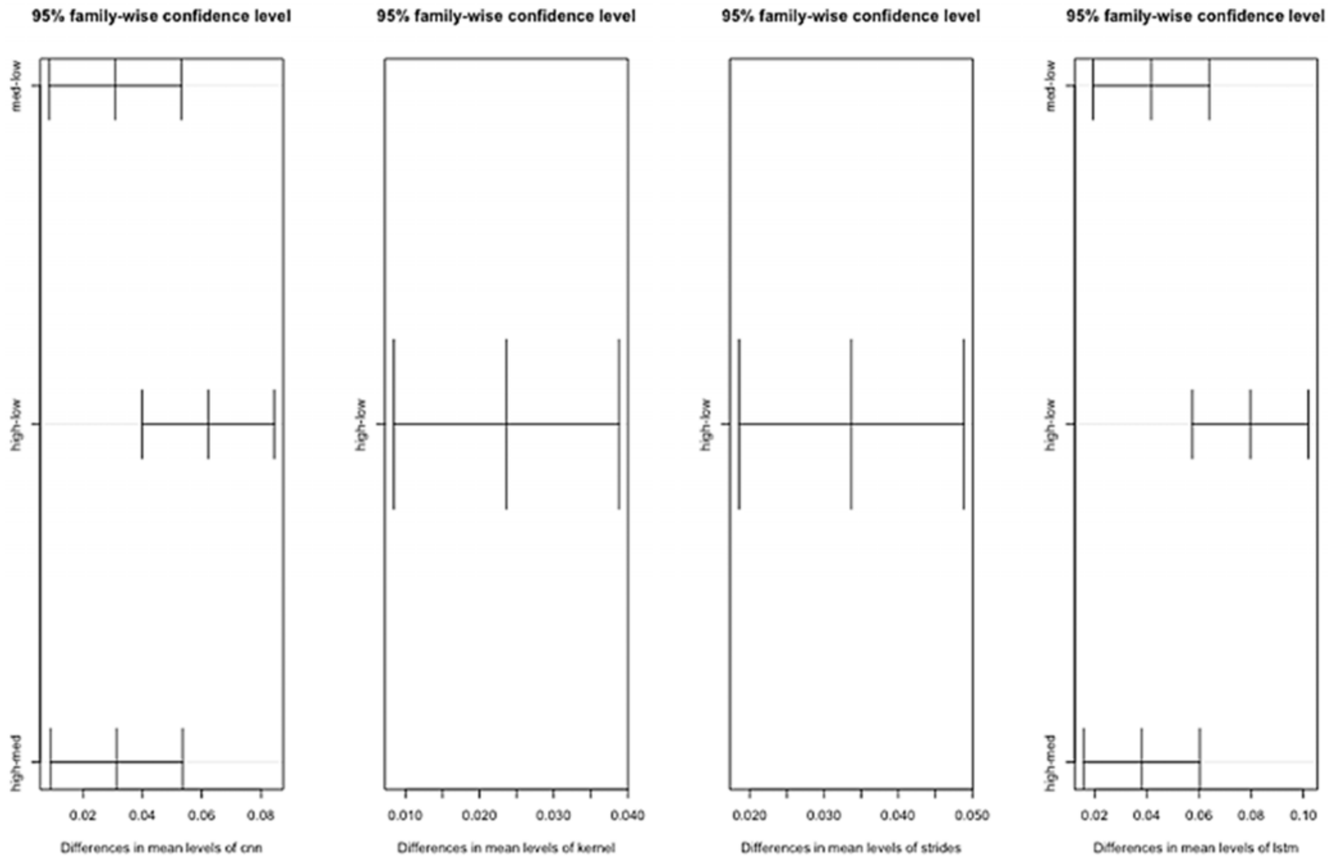

Fig. 2 Confidence intervals on the differences between the means

As observed from Fig. 2, 'high' levels for all statistically significant factors with positive differences in means provide the best levels for the hyperparameters. Thus, the best levels for number of filters in the convolutional layer, kernel size, strides and number of units in the LSTM layer are obtained as 128, 5, 2 and 64, respectively. A convolutional recurrent network was run with these settings to develop an author classification model using the training and validation data. As the dropout rate is not statistically significant, a value of 30% is used in developing this final model. The model performance is assessed by predicting the author of an article available in the test data. For the 50 authors in the test data, an average accuracy of 31.8% and median accuracy of 28% are achieved. Minimum and maximum author classification accuracies based on the convolutional recurrent network model were 8% and 90% respectively, which also highlight the challenging nature of this author classification task. These results are compared with a LSTM model with 64 units that is built using the same training and validation data. The LSTM model with the test data showed an average accuracy of 21.3% and median accuracy of 16% for correctly classifying authors. In addition, the LSTM model has a range from 4% to 94% in correctly classifying authors in the text data. Author classification accuracy for convolutional recurrent network and LSTM networks using test data are summarized in the form of boxplot in Fig. 3.

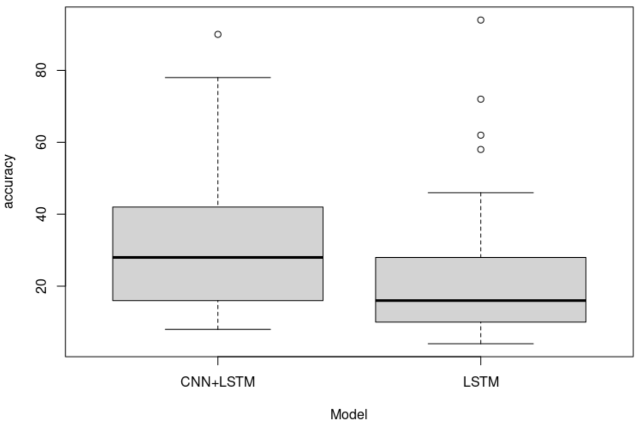

Fig. 3 Boxplots comparing accuracy of text classification

It is observed from Fig. 3 that the convolutional recurrent network model performs better when carrying out author classification compared to the LSTM network. The boxplot suggests median author classification based on convolutional recurrent network to be better than the third quartile for accuracy based on the LSTM model. In other words, when using the LSTM model 75% of the results are inferior to those obtained by median accuracy based on CRNN model.

VI. Summary, Conclusions and Future Work

This paper focused on developing a convolutional recurrent network deep learning model for author classification based on articles that they have written. Convolutional recurrent networks combine the advantages of two networks into one network. On one hand convolutional networks can capture high level local features from the text data, and on the other hand recurrent networks such as LSTM can capture long-term dependencies in the data involving sequences. Convolutional recurrent network first extracts features using 1-dimensional convolutional layer. These extracted features are then passed to the LSTM recurrent layer to obtain hidden long-term dependencies which in turn are passed to a fully connected dense layer. The dense layer in the model calculates the probability of correct classification for each author based on the article data. This paper emphasizes the use of planned experimentation to fine-tune the hyperparameters and achieve a model with high accuracy in author classification.

The results indicate that the convolutional recurrent network achieved a mean accuracy of approximately 31.8% in correctly classifying authors based on articles, compared to a mean accuracy of approximately 21.3% when only the LSTM network is used. This represents an improvement in mean accuracy of about 49%. The results obtained also highlight the challenging nature of problems involving text data and the related limitations of deep learning models. While the convolutional recurrent network in this paper was applied to the author classification problem, future work can explore the use of this type of deep network for applications in other domains that involve sequential data, such as natural language processing, speech recognition, and video related problems.

Acknowledgment

This research work was carried out using Dell's Mobile Data Science Work-station Precision 7750 with NVIDIA Quadro RTX 5000. I'll like to thank Dell and Nvidia for loaning this powerful laptop to support my deep learning and artificial intelligence related research.

References

[1] B. K. Rai, B. *Advanced Deep Learning with R: Become an expert at designing, building, and improving advanced neural network models using R*. Packt Publishing, Birmingham, 2019, pp. 253-276.
[2] A. Chavhan, S. Chavan, S. Dahe, and S. Chibhade, "A neural network approach for real time emotion recognition," *International Journal of Advanced Research in Computer and Communication Engineering*, 4(3), 259-263, 2015.
[3] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y.A. Nanehkaran, "Using deep transfer learning for image-based plant disease identification," *Computers and Electronics in Agriculture*, 173, 105393, 2020.
[4] S. Khan, N. Islam, Z. Jan, I. Din, and J. Rodrigues, "A novel deep learning-based framework for the detection and classification of breast cancer using transfer learning," *Pattern Recognition Letters*, 125, 1–6. Doi: 10.1016/j.patrec.2019.03.022, 2019.
[5] B. K., Rai, and A. Meshram, "Application of neural network to detect freezing of gait in patients with Parkinson's disease," chapter in book titled Soft Computing, edited by Mangey Ram and S. B. Singh, De Gruyter, 2020.
[6] M. Tounsi, I. Moalla, F. Lebourgeois, and A. Alimi, "CNN based transfer learning for scene script identification," in *International Conference on Neural Information Processing*, pp. 702–711, Springer, Cham. Guangzhou, China, 2017.
[7] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ArXiv* Preprint ArXiv:1409.1556, 2014.
[8] P. Tiwari, and A. Darji, "A Novel S-LDA Features for Automatic Emotion Recognition from Speech using 1-D CNN," *International Journal of Mathematical, Engineering and Management Sciences*, 7(1), 49-67, 2022.
[9] A. Jacovi, O. Shalom, and, Y. Goldberg, "Understanding convolutional neural networks for text classification," arXiv preprint arXiv:1809.08037, 2018.
[10] J. Zhao, X. Mao, and L. Chen, "Speech emotion recognition using deep 1D & 2D CNN LSTM networks," *Biomedical Signal Processing and Control*, 47, 312-323, 2019.
[11] A. Bhunia, A. Konwer, A. Bhunia, A. Bhowmick, P. Roy, and U. Pal, "Script identification in natural scene image and video frames using an attention based convolutional-LSTM network," *Pattern Recognition*, 85, 172–184, 2019.
[12] S. Lyu, and J. Liu, "Convolutional Recurrent Neural Networks for Text Classification," *Journal of Database Management (JDM),* 32(4), 65-82, 2021.
[13] C. Zhou, C. Sun, Z. Liu, and F. Lau, "A C-LSTM neural network for text classification," arXiv preprint arXiv:1511.08630, 2015.
[14] D. Yogatama, C. Dyer, W. Ling, and P. Blunsom, "Generative and discriminative text classification with recurrent neural networks," arXiv preprint arXiv:1703.01898, 2017.
[15] L. Guo, D. Zhang, L. Wang, H. Wang, and B. Cui, "CRAN: A Hybrid CNN-RNN Attention-Based Model for Text Classification," in: Trujillo J. et al. (eds) Conceptual Modeling, Lecture Notes in Computer Science, vol 11157. Springer, Cham, 2018.
[16] J. Allaire, and F. Chollet, *Deep Learning with R*. United States: Manning, 2018.
[17] R. G. Miller, *Simultaneous Statistical Inference*. Springer, 1981.
[18] B. S. Yandell, *Practical Data Analysis for Designed Experiments*. Chapman & Hall, 1997.

Bharatendra Rai is PhD in industrial engineering from Wayne State University, Detroit in 2004. He has master of technology degree in quality, reliability, and operations research from Indian Statistical Institute, India in

1993, and another master's degree in statistics from Meerut University, India in 1991.

He is currently chair and professor in the department of decision and information sciences at Charlton College of Business, University of Massachusetts-Dartmouth. He has earlier worked as a quality and reliability engineer at Ford Motor Company in Detroit. He has authored a book titled 'Advanced Deep Learning with R" published by Packt. He has also co-authored another book titled 'Reliability Analysis and prediction from Warranty Data: Issues, Strategies, and Methods' (New York, NY: CRC Press Taylor & Francis Company, 2009). He is author of over 50 journal and conference articles including publications in IEEE Transactions on Reliability.